



(12) **EUROPEAN PATENT APPLICATION**

(43) Date of publication:
21.08.2002 Bulletin 2002/34

(51) Int Cl.7: **G06F 9/445**

(21) Application number: **01410021.8**

(22) Date of filing: **19.02.2001**

(84) Designated Contracting States:
AT BE CH CY DE DK ES FI FR GB GR IE IT LI LU
MC NL PT SE TR
Designated Extension States:
AL LT LV MK RO SI

(71) Applicant: **Hewlett-Packard Company**
Palo Alto, CA 94304 (US)

(72) Inventors:
• **Raffaele, Eric**
38000 Grenoble (FR)

• **Brebner, Gavin**
38410 Saint Martin d'Uriage (FR)

(74) Representative: **Lloyd, Richard Graham**
Intellectual Property Section,
Legal Department,
HEWLETT-PACKARD FRANCE,
Etablissement de Grenoble
38053 Grenoble Cedex 9 (FR)

(54) **Process for executing a downloadable service receiving restrictive access rights to at least one profile file**

(57) A process for executing a downloadable service with specific access rights to at least one profile file in a user's computer. The web browser is used for accessing a web server through a first set of communication port and socket. Therefore is arranged a confined runtime environment such as for example an extended sandbox (11) and used in Java programming which is assigned a second communication port and sockets and

which is used for executing downloadable service with restricted access on at least one profile file. The process can be used for executing services under the form of a signed archive file which signature is used for both validating the archive file and for selecting one particular secure class loader prior to the generation of the compiled code of the service.

Description

Technical field of the invention

[0001] The invention relates to communication systems and more particularly to a process for executing a downloadable service receiving restrictive access rights to at least one profile file.

Background art

[0002] The constant progress of the Information Handling System (I.H.S.) technology and that of the communication systems, particularly with the explosion of the Internet and intranet networks, have resulted in the development of an era of information and services. With a single personal device, a user can get a connection to the Internet network, and have direct access to a wide range of information and services, including electronic business applications.

[0003] Solutions were developed in the art for rendering easier the access to these huge amounts of resources and for improving the possibilities of negotiation and new services. For this purpose, agents were developed for allowing the customers to delegate some specific tasks to automatic modules in order to simplify part of the searching process.

[0004] US 5,877,759 assigned to Netscape Communications Corp. and entitled *Interface for user/agent interaction* shows a first example of a user interface which provides assistance to the client user and an interface that is centred on autonomous processing of whole tasks rather than sequences of commands.

[0005] International application number WO 98/47250 assigned to IBM Corp. and entitled *Apparatus and method for communicating between an intelligent agent and client computer process using disguised messages* illustrates an agent for the communication with a client and which uses a selector known by both parties to generate and interpret messages and thereby to effectively disguise confidential information transmitted in the message from third parties.

[0006] International application number WO 98/43146 assigned to IBM Corp., and entitled *Intelligent agent with negotiation capability and method of negotiation therewith* is another example of an improvement brought to a so-called intelligent agent.

[0007] In addition to the development of agents, the use of one or more profile files located within the user's machine, is clearly a very promising field of development. A profile file is usually more easily updated since the customer feels more concerned about the information that he loads into his own profile file(s). Further this information continuously remains under his/her own control. When the customer permits it, the use of the information loaded into that or those profiles shows to be a material of the highest interest for service providers since it clearly enhances the personalization and the

possibilities of negotiation with the customer. Practically, the profile files can be used by product/service providers who wish to propose an offer well tailored to the customer's needs.

[0008] Unpublished European patent application n° 0041004.6 entitled *"process and apparatus for allowing transaction between an user and a remote server"* by G. Brebner, filed on 21. January 2000 and assigned to the Assignee of the present application, shows a profile file which is used for improving the possibilities of negotiation between a customer and a product or accessories provider. Basically, there is disclosed a process which receives an abstract request expressed by the user. The request is subject to contextual analysis for the purpose of preparing an aggregate request based on the abstract formulation of the client, which can then be completed with additional technical information which is extracted from a local profile. Since the products/service providers are automatically given direct access to the internal configuration of the machine, of which the user may even not be aware, they can propose offers which are correctly tailored to the users, even when those are not versed in the field of computers.

[0009] While the use of one or more profiles substantially enhances the personalization of the access to the web, as well as the possibilities of negotiation between the customer and the service/product providers, it can be seen that the customers feel more and more concerned with the question of the privacy of their private data, and the risk of any misuse of that information.

[0010] Mechanisms have been developed for minimising the exposure of sensitive data on the Internet and particularly the highly confidential information that is contained within such profile files. Unpublished European patent application n° 00410005.3 (HP50001003) entitled *"method for a personalised access to the Internet network"* by G. Brebner et al, filed on 21. January 2000 and also assigned to the Assignee of the present application brings a first solution to the problem of privacy. There is disclosed a process, still based on an agent, which avoids the dissemination of the user's personal data over the web. For that purpose, an agent located on the user's machine generates a Hyper Text Markup Language (HTML) page for the user and which takes into account private data contained into at least one profile file. At different predetermined intervals, the Agent module polls one or more predetermined service provider(s), and downloads a file that simultaneously contains both data representative of products or services, and rules for stating the conditions for the attribution of the data. By comparing the rules with the private data contained within said at least one profile, a local data mining process is achieved and the Agent can construct a personalised HTML page which contains an offer well tailored to the customer's needs. This can be finally displayed into the web browser of the client machine. Since the HTML page is locally generated with the private data that is contained in the profile(s), and which belongs to

the user, the service provider does not need to create and maintain any kind of centralised database, and the user profile remains in the user's machine.

[0011] The present invention extends the possibilities of exploitation of the profile files, which substantially increases the security of the access to the important information therein contained.

Summary of the invention

[0012] It is an object of the present invention to enhance the flexibility and the use of profile files containing highly sensitive information, while ensuring a high degree of security in respect of that information.

[0013] This object is achieved by the present invention which provides a process for executing a downloadable service within the client machine which receives very restricted access rights to the user's profile file(s).

[0014] Basically, there is provided a process which involves the step of accessing a server on the Internet or an intranet network through a web browser which is assigned a first port and socket for the communication with the web server. When the user requests the downloading of an executable service from a web server, the web browser launches confined run time environment which is assigned a second port and socket for receiving the downloadable service. The confined run time environment is arranged to provide restrictive access rights to at least one profile file(s), so that the executable service may only access the profile file(s) through the methods and access rights which are defined by the confined run time environment. This substantially minimize the risk of any misuse of the valuable information contained with the user's profile.

[0015] Preferably the confined run time environment corresponds to an extended sandbox in the context of the java programming. The personal service is received under the form of set of data (remote information, sorting code, requested data) which reference is contained within an Hyper Text Mark-up Language (H.T.M.L.) page accessed by the web browser. The service is received in an archive file, such as a jar file for instance, which is signed and which contains a class structure of java code. The java code may also be received from a File Transfer Protocol (FTP) server.

[0016] In one embodiment, the process validates the archive file and in accordance with the signature existing within the determined archive file, uses a predefined jar and secure class loader for embodying a specific corresponding sandbox associated to that class structure. Therefore there is achieved a direct association between the signature of the applet file and the java security mechanisms which will be used for executing the service.

[0017] Preferably, the predefined jar loader is used for controlling the access to at least one profile file so that the service which arrives within the extended sandbox and which is compiled in accordance with the predeter-

mined jar loader receives a personalized access to said profile(s) file. There is therefore achieved a high control and security on the information belonging to the user.

[0018] More preferably, the predefined personalised sandbox is used for providing methods allowing a selective access to some specific areas or sections of the profile file(s) which contains an hierarchical structure of data, user display, and remote communications. In one application, the profile file includes personal data belonging to the user, while an additional section comprises technical data concerning his/her computer.

[0019] When the signed archive file is received and opened with the corresponding jar loader, the latter implements methods, according the user policy, to the already contained classes for the purpose of personalizing the access rights to the profile. The java service code can then be executed in its personalized environment and starts processing tasks, possibly on the personal data of the user. A local service can then be executed on the local machine with some of the user's private data. As soon as the result of the service is made available, the latter can be transmitted back to the web server for additional process and negotiation without requiring the exchange and transmission of the private data of the user loaded within the profile files.

[0020] In one embodiment, the service is used for executing a data mining process on the information loaded into the profile(s).

[0021] The process according to the invention increases the possibilities of personalization of the access for service providers since different jar loaders may be used for different signatures and different service providers. Additionally, there is provided a wide range of new high added-value services for the customers.

[0022] Even in the case of one unique jar loader, the latter can be used to arrange a set of different access rights to some specific areas or sections of the profile(s).

[0023] A powerful downloadable service can be executed within the local machine of the customer while minimising any risk of misuse or abuse of the user's private data. The validation process based on the signature and, additionally, the access of the profile through the restrictive access permissions arranged by the sandbox and the class loader substantially secures the use of the user's profile.

[0024] In one embodiment, when the archive file of the service is received within the extended sandbox, the service is being executed and initiates a processing task which may continue, even if the connection with the web server is being discontinued. When the task completes and the result is made available, the latter will be transmitted to the web server at the next dial-up and connection for the purpose of a further processing of the negotiation.

[0025] In some extent, there is achieved the concept of "consumable" service - requiring no installation process as for the other software packages - which can be downloaded in an archive file and received in a very per-

sonalized environment, with restrictive access to the profile file of the user. When the execution of the service completes, the useful result is provided to the user or to a web server and the service is discontinued. A wide range of applications and new added-value services can use the process for the benefit of the customers, such as travel or flight booking service, electronic business, data mining and book-keeping processing tasks.

Description of the drawings

[0026] Embodiments of the invention will now be described by way of examples only, with reference to the accompanying drawings, wherein:

Figure 1 illustrates the basic principle of the transmission of a downloadable service into a local client computer.

Figure 2 is a flow chart illustrating the process which is executed when a service is downloaded, received and executed within the extended sandbox.

Figure 3 is a flow chart illustrating the different messages being exchanged between the web browser, the extended sandbox and the web servers.

Figure 4 illustrates the architecture of one preferred embodiment of a confined run time environment.

Description of the preferred embodiments of the invention

[0027] As will be shown below, the negotiation process is handled by means of a temporary or consumable service which is launched within the existing confined run time environment and which receives some access rights to the user's profile. An example will be considered with a travel agency service which may take advantage of the useful and valuable information loaded within the user's profile file, such as his personal options, the number of children, etc.

[0028] The preferred embodiment will be described with a service which is downloaded under the form of a jar file contained within an Hyper Text Mark-up Language (H.T.M.L.) page. Clearly, this is only an example of an implementation and other embodiments could be contemplated.

[0029] As shown in figure 1, a user's computer 1 is connected to a communication network 2, such as an Internet or Intranet network for instance. In this way, he may access to multiple different service servers, and particularly to an electronic service portal 3 which is prepared to accept and process communication as will be described hereinafter with details. A web server 4 - for instance a travel agency portal - may be dedicated to receive air-flights booking or reservation commands.

[0030] The user's computer 1 - or client - includes a

web browser 15 which is, for instance, *Internet Explorer* 4 or 5 (manufactured by Microsoft Corp.) or *Netscape Explorer* (manufactured by Netscape Communications Corp. The web browser is used for accessing HTML pages located from web server 3 and 4 and, when appropriate, for downloading the applet therein included.

[0031] As known in the field of Java programming, the expression sandbox refers to a confined runtime environment. These are sometimes also called rubber-walled "jail" environments and are notional spaces within a computer in which certain programs can run in isolation from the host system. This can also refer to the safe mode of the Java Virtual Machine which turns on when a Java applet is run inside a web browser. Such an environment can be used to run untrusted Java applets that are downloaded from the web. Practically, a sandbox is a set of limitations or restrictions which are imposed on untrusted applets and are designed to prevent those applets from acting so as to compromise the client system. This is done by disallowing access to certain APIs by means of security exceptions.

[0032] An extended sandbox 11 is launched when a service is to be received by the client machine, as will be described hereinafter with further details. The extended sandbox is arranged for organizing a restricted access to the user's profile file. As known by the skilled man in java programming, the security is based on four distinctive elements forming the sandbox: the Byte Code Verifier, the Secure Class loader, the Access Controller, and the Security Manager. Together, these four elements perform load and run time checks to restrict file system and network access, as well as access to browser internals. More information can be found on the Java Security Web Site which address is provided in <http://www.rstcorp.com/java-security.html>

[0033] The profile file is used for gathering, when appropriate, the user's personal data (such as preferences and knowledge) possibly combined with technical information concerning the environment of the user's machine, user's compartments, habits and preferences. As most users are not versed within the field of computers, it can be advantageous to organize one particular section of the profile file with technical information which is automatically collected by system service 12 of figure 1. System service 12 is based on a so called *sysinfo.exe* executable file which extracts system information directly from the SMBIOS tables, or interrogates the Distributed Management Interface (DMI, or Windows Management Instrumentation (WMI) as known from Microsoft. As known by the skilled man, the DMI interface is an Application Programming Interface (API) that consists of a set of routines that are called for accessing the information stored within the BIOS layer. Basic information relating to the DMI programming interface can be found at the address <http://www.dmtf.org/spec/html>. By using the DMI or WMI interfaces, or by accessing directly the SMBIOS level, the *sysinfo.exe* executable file accesses the different tables contained in the System

Management BIOS (SMBIOS) for the purpose of reporting comprehensive information regarding the user's configuration, and required for completing a request for transaction. Such information includes the type of processor, the type of chipset, the number of hard disk drives, the particular graphic card being used, the serial number of the display, the reference of the operating system and so on. User related profile can be collected using explicit techniques such as specific forms, or using implicit techniques such as cookies, spyware, click stream tracking, or a mix of these techniques.

[0034] In one embodiment, the profile file(s) can be arranged in a hierarchical organization, with different sections containing information of different category. The profile file is embodied under the form of a known Extensible Mark-up Language (XML) standard which is defined by the World Wide Web Consortium (W3C) and which is associated to Document Type Definition (DTD) file.

[0035] In one embodiment, the profile may comply with the recommendations provided by the CPEX (Customer Profile EXchange). This consortium works to realise a specification in order to define a data model for profile information that relate customers of enterprises. The specification also defines metadata to associate privacy control on customers information, and a protocol for query, deliver and update these information. CPEX refers to other standard such as XML, P3P, LDAP/DSML, AUO.

[0036] With respect to figure 2 there will now be discussed the basic process which is executed for launching the consumable or temporary service when the user is surfing over the Internet.

[0037] In a step 21, the user establishes a connection to the electronic service portal 3 by introducing the appropriate Uniform Resources Locator (U.R.L.) of the web portal, such as, for instance, <http://www.hp.eservice.com>. Clearly, the communication can take advantage of the wide generalisation of the Internet protocols, and in particular the Hyper Text Transfer Protocol (HTTP). In the preferred embodiment, the local agent issues a request for transaction which is embedded into a HTTP GET request which is recognised at every level, and particularly by every Proxy. The particular format of the HTTP GET request is defined in the well-known rules laid down in the Request For Comments (RFC) 2.6.1.6, which are available at the following address <http://www.w3.org/protocols>.

[0038] The web portal receives the request and prepares the construction of an Hyper Text Markup Language (H.T.M.L.) page which is transmitted back to the user's computer and pushed into his web browser 15. In one embodiment, the process may take advantage of the teaching of the above mentioned European patent application n° 0041004.6, which is herein incorporated by simple reference, for the purpose of completing the abstract user's request with some basic information extracted from the profile. In that situation, the user is al-

lowed to enter only partial information - referred to in the following as an abstract request since it only contains some basic incomplete information - and process it for completing the transaction.

[0039] In step 22, the user selects a service that causes the transmission of a corresponding request to the service portal 3. In the preferred embodiment the service is downloaded under the form of a package file (such as a jar file). This download launches a specific sandbox which is arranged to communicate via a port and socket which may be different to that used for the normal HTTP connection by the browser 15. As known by the skilled man, the usual port is port 80 and the sandbox can conveniently be arranged to listen for communication on another available port.

[0040] In step 23, the web server responds by transmitting an archive file that is received by the extended sandbox on its dedicated port and socket. The archive file is a signed file, for instance jar file which contains a set of class structure corresponding to the java code classes of the service being desired. It is clear that the process is directly applicable to java code classes but should be understood as applicable for other types of data structures, such as agent technology.

[0041] In step 24, the corresponding jar loader is selected using the corresponding transmission port of the service portal 3. The jar loader starts with step 25, i.e. checking the signature and validates them in accordance with the known java security mechanisms. In the case where the signature is corrupted, the service code is discarded by the java security mechanisms. The processing of signatures and the security manager depends upon the particular web browser being used. Such mechanisms are well known by the skilled man and will not be further elaborated on. More information on java security mechanism may be found in the following references, *Java Security* from Scott OAKS, Edition O'Reilly and *Java Cryptography*, from J. KNUDSEN, Ed. O' Reilly.

[0042] In step 25, the signature is used for selecting one particular secure class loader among a set of predefined class loaders, and permission in order to control and isolate the service application. The secure class loader is used for opening the jar archive file, for generating the classes of the java code. By choosing a selected class loader, the process causes a direct association of the security policy with the originator of the service being downloaded. It can be seen that the signature is used for checking the integrity of the file and, also, for choosing a particular security policy mechanism.

[0043] In step 26, the archive file (.jar) is opened and the class loader that was selected in step 25 generates the set of classes contained within the service, and links interfaced methods to the corresponding implemented sandbox methods. Therefore, it can be seen that the conventional Application Programming Interface (API) which is used for generating the sandbox upon the reception of the set of java classes is modified for the pur-

pose of processing the java classes coming from a determined service provider. Therefore, a set of selective and personalised permissions and access rights can be associated to some services received from some service providers. The set of permissions is used for providing a selective access to at least one profile file, for instance a profile containing user private data.

[0044] In step 27, the corresponding java code is being compiled with the appropriate access rights to the user's profile. These rights allow or refuse the access for the missing data to personalize the service. Then, in step 28, the code used to sort the remote information is executed within the extended and personalized sandbox.

[0045] When the result of the process is available, it is transmitted back in step 29 to the user and/or web server 3 or to a different web service provider 4. The web service 3 or 4 can then prepare a well-tailored offer which will be transmitted back to the web browser 15 of the client.

[0046] It can be seen that a highly personalised transaction is achieved in a personalized and secured environment. The signature is used for validating the applet being downloaded and, additionally, servers for personalizing the access rights to be assigned to the applet. The access rights can be defined in a very flexible way, simply by introducing new java classes in the compile code by means of the predefined jar loader. Therefore, a given service for example, corresponding to a flight-booking service may be authorized to get access to the user's profile, and particularly to the private section summarising the composition of the users family, the different options which are usually favoured by the family etc... On the other hand, the service will not be allowed to access other part of the profile relating to medical or professional sections.

[0047] The personalization is rendered very flexible since one single jar loader may provide precise and full control over the different branches and sections of the user's profile. The use of such a profile by web providers will significantly enhance the access to the Internet.

[0048] Figure 3 illustrates the different messages which are exchanged by the different elements in the negotiation process. Upon request of the user, as referenced by arrow 31, the browser transmits a request to web server 3 for the purpose of downloading an appropriate archive file. The web server 3 gathers a set of information, illustrated by arrows 32 and 33, asking service provider throw the web. The web server 3 creates an archive file and transmits it to the user's extended sandbox 11, as illustrated by arrow 34. The sandbox executes the downloaded service which access the private data loaded within the user's profile, and when the result is available, it is transferred to the user browser in order to wait his final choice as shown arrow 35. One time the user final choice done, the result can be transmitted either first to the service portal 3 (arrow 361), alternatively to another web server (not shown), next to

the service provider 4 (arrow 362), either directly to the service provider (arrow 37). The communication with the service portal 3 and/or the service provider 4 is done if it is appropriate. The service provider can answer, if necessary (e.g. to confirm a booking), via an HTML page sent either first to the service portal 3 (arrow 381) next to the user's browser 15 (arrow 382), either directly to the user's browser 15 (arrow 39). In this confirmation phase, the service provider 4 can be useless, in this case arrows 362, 37, 381 and 39 are not used.

[0049] The exchange of messages described above shows how it is easy to personalize the communication between the user and one or more service provider, while securing the use of the private data loaded within the user's profile file.

[0050] In the following description, there will be described a specific structure of a profile file and the example of the access to that profile file.

[0051] The kind of access to a profile depends on the context of the use. Data are created, updated, deleted and consulted, according to a use. In this way, it is not always necessary to provide the all set of personal data to a user or a service. Contexts have to be created.

[0052] According to this, when a service requests for data in order to personalise itself, it has to link its requests to a context. For instance, a "travel booking service" can ask for number of children but a "grocery buying online service" does not need to ask such a question. Asking a credit card number is justified by a final booking phase but can be allowed for a simple data research. It is the aim of the profile access user's permissions to control, enable and disable the access to the user's personal data.

[0053] With respect to figure 4, there is shown a preferred embodiment of a user platform in a java environment. The Personal Personalised environment 410 comprises a jar loader 420 which is linked to the personal sandbox 430. Three interfaces, namely a remote exchanges interface 440, a profile interface 450, and a user interface 460 are used for exchanging data with other parts of the system. In the jar file there are three types of data which are used for embodying the service: Imported Data 470; Service Code 480 and Requested Data 490. The imported data 470 are provided by the service providers and describes the offers which are available. The service code 480 describes how to match or personalize the specific offer made to the user, based or using the imported data filtered by elements taken from the user's profile, i.e. the Requested Data 490. In order to access the personal data stored in the user's profile, the Service has to access the Profile Interface 450 which implements the specific context security policy. This is achieved by using a set of signatures (not shown). In Requested Data 490, a method such as "ask for" will request a name in the given context, for instance "travel booking". In order to interact with the user, the service has to use a User interface 460 which implements the appropriate functionalities for the device.

Clearly the complexity of the user interface depends upon the particular device being involved, for instance a Personal Digital Assistant (P.D.A.), a personal computer, a mobile telephone, or even an Internet watch.

[0054] In the particular case where some results of the execution of the service needs to be transmitted back to a service provider, the service communicates via the Remote Exchange Interface 440 which limits the communication and the kind of information which is reported back to the service provider. This is clearly an additional advantage since it provides a high level of flexibility in the control of the data which is received and analysed, including the full control on the use of the results of that analysis which is possibly reported back to the service provider.

[0055] Any type of service could take advantage of the architecture shown in figure 4. In the case of travel booking service, for instance, the process executes the steps which were explained in detail in reference with the general flow charts of figure 2 and 3. In the case of travel booking, the user accesses the web portal (step 21 of fig. 2) of one travel service provider and requests the booking of a particular destination and date. The travel service provider then constructs a package to reply to the request which consists of offers appropriate to that request, code which, when executed, will personalize the choice and an indication of the data required from the user's profile. This is the archive file received by the user's device in the form of a jar archive file as illustrated in step 23 of figure 2.

[0056] The jar archive is loaded by the Jar Loader as shown in step 24 and isolated immediately. This is achieved by means of the Personal sandbox shown in figure 4. As known by the skilled man, the Personal sandbox uses the text file embodying the Global Services Security Policy. It should be noticed that, in the preferred embodiment, the particular GSSP which is used allows no access on the user's resources or communication with the service provider. This is clearly an advantage because the user keeps the control on the information which is likely to be reported back to the Service provider.

[0057] The signature is used to identify the origin of the service and to associate a dedicated security policy in accordance with the user's privacy policy. To achieve this, the profile interfaces gets the contents of a text file containing the Context Security Policy which closely depends on the particular service being considered.

[0058] According to this specific security policy, the service asks for the requested data which are stored within the profile. The profile interface, implementing the controlled methods to access the profile, passes or blocks the data access.

[0059] The service code is then executed, using the combination of imported data provided by the service provider - for instance the choice of a destination of the flight to London plus a departure time as well as additional data giving details of the offers matching the orig-

inal request), and the data taken from the user's profile.

[0060] The description above illustrates with what ease the architecture shown in figure 4 permits to embody any kind of service for enhancing security in the user's profile. Clearly, the architecture could even be used for embodying an authentication service arranged to cooperate with a smart card interface so as to permit validation of a service in accordance with the code typed by the user. Because only the service is being transmitted over the Internet network the security of the personal data belonging to the user is substantially increased.

[0061] Although the invention has been described by way of example and with reference to particular embodiments, it is to be understood that modifications and/or improvements may be made without departing from the scope of the appended claims.

[0062] Where in the foregoing description reference has been made to integers or elements having known equivalents, then such equivalents are herein incorporated as if individually set forth.

Claims

1. A process for executing a downloadable service with specific access rights to at least one profile file in a user's computer, said computer comprising a web browser communication to the Internet or intranet via a first communication port and socket, said process being **characterized in that** it involves the steps of:
 - arranging a confined run time environment (11) which is assigned a second communication port and socket and providing restricted access to at least one profile file;
 - downloading said service through said second communication port so that it is received by said confined run time environment (11); and
 - executing said service within said confined run time environment whereby said service is given an access to said at least one profile file in a secure environment.
2. The process according to claim 1 **characterized in that** said confined run time environment is an extended sandbox having restrictive access to said at least one profile file.
3. The process according to claim 2 **characterized in that** the service is downloaded under the form of a set of java code containing classes structure packaged within a signed archive file. Three parts compose the service: remote Internet data, a list of requested data that are needed to personalise the service, and code to sort remote Internet data using requested accessible data.

4. The process according to claim 2 **characterized in that** it further involves the step of:
- downloading a signed archive file which is received via said second communication port;
 - checking the signature of the archive file and, if corrupted, discarding said archive file; and
 - selecting one particular secure class loader corresponding to said signature for the purpose of creating additional security java classes allowing the restricted access to said at least one profile file.
5. The process according to claim 4 **characterized in that** said secure class loader further includes restricted access to some specific sections of said at least one profile files.
6. A process for receiving a signed archive file containing class structures representative of at least one service to be downloaded to and executed on a client computer, **characterized in that** said at least one service is associated with a corresponding set of access rights to some profile files and the process further involves the steps of:
- receiving said archive file;
 - validating said archive file with said signature;
 - selecting one secure class loader associated with said signature, said class loader being representative of the set of access rights associated to said service;
 - generating said classes in accordance with the secure class loader being selected for the purpose of generating a compiled code; and
 - executing said compiled code.
7. A process for generating compiled executable code in a client machine, comprising the steps of:
- receiving a signed archive file containing classes of a service to be executed on said local machine;
 - checking and validating the signature associated to said archive file;
 - associating said signature to one predetermined secure class loader for the purpose of defining a predetermined java security policy and assigning access rights to at least one profile file;
 - invoking generating said access control classes in addition to the classes of said service; and
 - generating compiled code and executing said compiled code.
8. The process according to claim 1 wherein said at least one profile is arranged in a hierarchical manner with a technical section comprising data representative of the user's machine configuration, user's compartments, habits and preferences.
9. The process according to claim 8 wherein said technical section is automatically filled with data extracted from information available at the Basic Input Output System (BIOS) level and profile building tools.
10. The process according to claim 8 wherein said technical section is automatically filled by means of interrogating standardised systems management interfaces present in the client computer.
11. The process according to claim 8 wherein said technical section is automatically filled by means of interrogation via the Distributed Management Interface (DMI) or Window Management Interface (WMI).
12. The process according to claim 1 wherein said downloadable service is an authentication service cooperating with a master card.
13. A transaction aid for assisting a transaction between an user and at least one remote server (3, 4), said transaction aid comprising program code elements for carrying out a process as claimed in any preceding claim.
14. A transaction aid as claimed in claim 11 in the form of a personal computer, the program code elements being implemented as a downloadable service having access to said at least one profile file.
15. A transaction aid computer program product having program code elements for carrying out a process as claimed in any of claims 1 to 9.

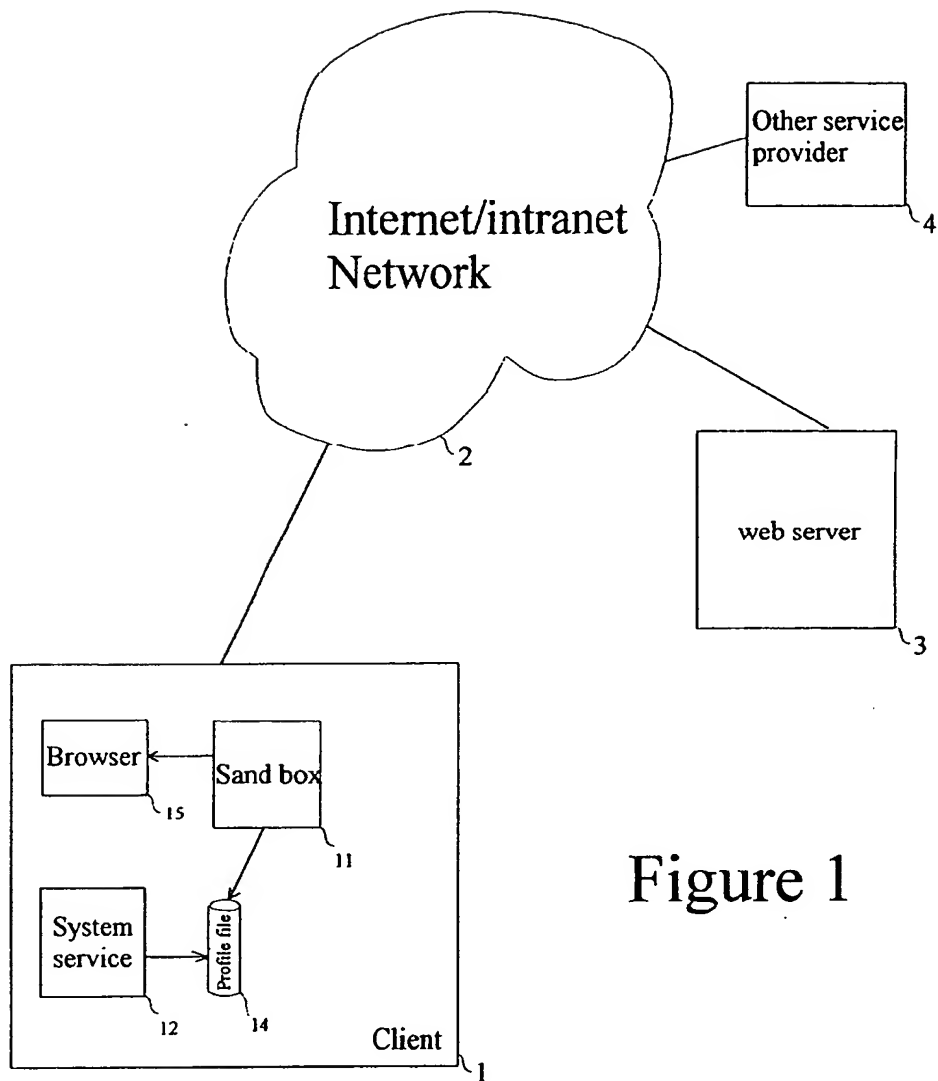


Figure 1

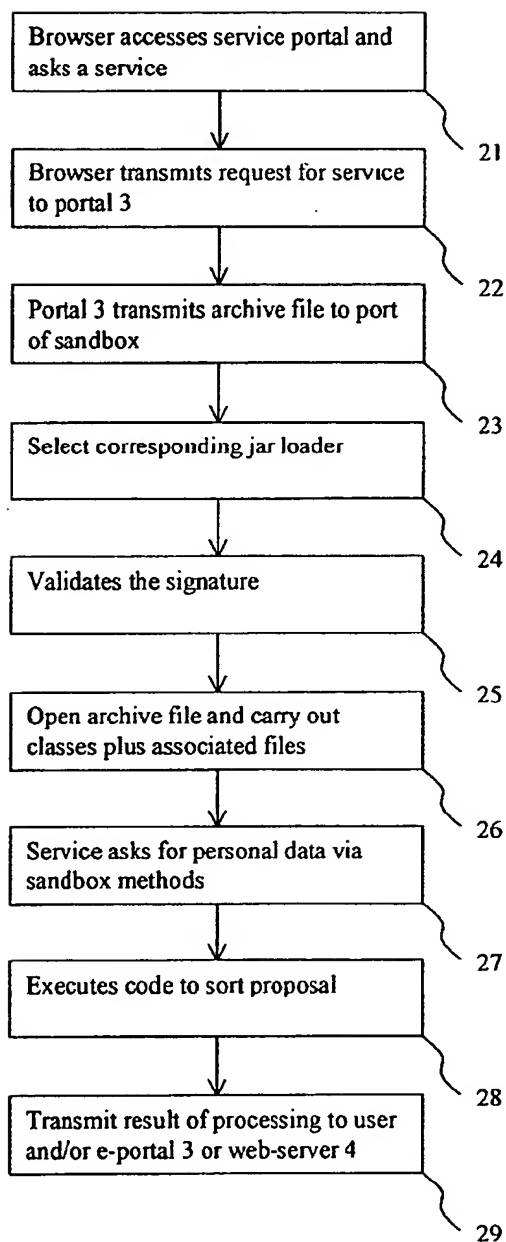


Figure 2

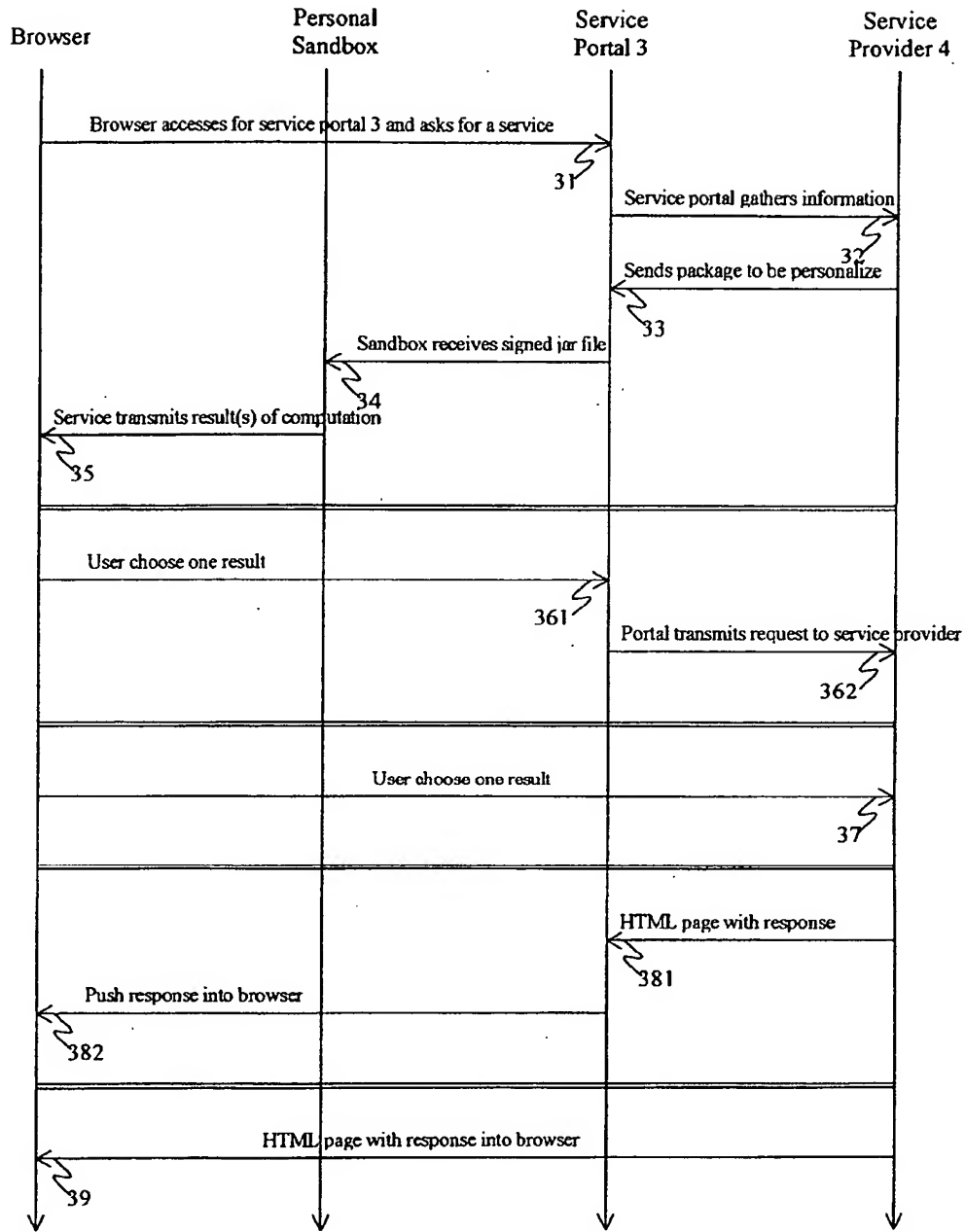


Figure 3

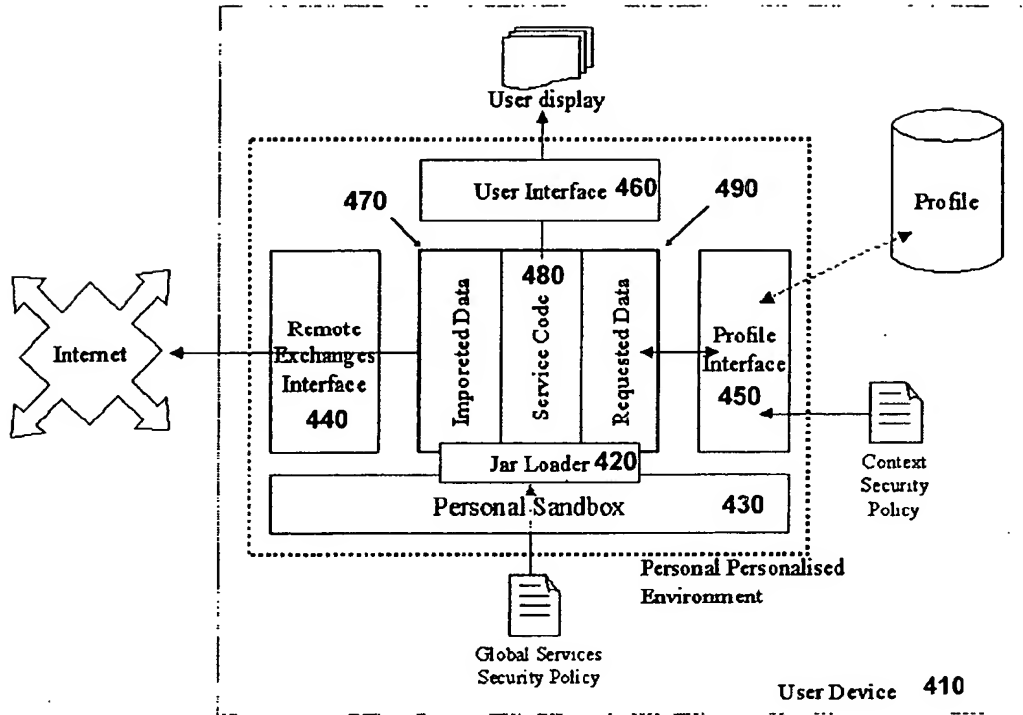


Figure 4



European Patent
Office

EUROPEAN SEARCH REPORT

Application Number
EP 01 41 0021

DOCUMENTS CONSIDERED TO BE RELEVANT			
Category	Citation of document with indication, where appropriate, of relevant passages	Relevant to claim	CLASSIFICATION OF THE APPLICATION (Int.Cl.7)
X	VENNERS B.: "SECURITY AND THE CLASS LOADER ARCHITECTURE" JAVAWORLD, IDG COMMUNICATIONS, 'Online! September 1997 (1997-09), pages 1-7, XP002171444 Retrieved from the Internet: <URL:http://www.javaworld.com/javaworld/jw-09-1997/jw-09-hood_p.html> 'retrieved on 2001-07-09!	1,2,4,6, 7,13,15	G06F9/445
Y		5,8	
A	* page 1, line 17 - page 2, line 2 * * page 2, line 16 - page 3, line 35 * * page 5, line 29 - line 44 * ---	3,9-12	
X	GONG L.: "JAVA SECURITY ARCHITECTURE (JDK 1.2). Version 1.0." , 'Online! 6 December 1998 (1998-12-06), pages 1-62, XP002171445 Retrieved from the Internet: <URL:http://java.sun.com/people/gong/papers/jdk12spec.pdf> 'retrieved on 2001-07-09!	1,2,4,6, 7,13,15	
			TECHNICAL FIELDS SEARCHED (Int.Cl.7)
A	* paragraph '01.2! * * paragraph '04.1! - paragraph '04.4! * * paragraph '05.1! - paragraph '05.4! * * paragraph '07.1! - paragraph '07.2! * ---	3,5,8,12	G06F
Y	WO 99 16003 A (BELARC INC) 1 April 1999 (1999-04-01)	5,8	
A	* abstract * * page 11, line 22 - page 13, line 3 * * page 14, line 25 - page 15, line 24 * * page 17, line 12 - line 20 * ---	1,2,4, 9-11,14	
	-/--		
The present search report has been drawn up for all claims			
Place of search THE HAGUE		Date of completion of the search 13 July 2001	Examiner Archontopoulos, E
CATEGORY OF CITED DOCUMENTS X particularly relevant if taken alone Y particularly relevant if combined with another document of the same category A technological background O non-written disclosure P intermediate document		1: theory or principle underlying the invention E: earlier patent document, but published on, or after the filing date O: document cited in the application L: document cited for other reasons S: member of the same patent family, corresponding document	

EP 01 41 0021 A1



European Patent
Office

EUROPEAN SEARCH REPORT

Application Number
EP 01 41 0021

DOCUMENTS CONSIDERED TO BE RELEVANT			
Category	Citation of document with indication, where appropriate, of relevant passages	Relevant to claim	CLASSIFICATION OF THE APPLICATION (Int.Cl.7)
A	US 5 870 544 A (CURTIS BRYCE ALLEN) 9 February 1999 (1999-02-09) * abstract * * column 6, line 31 - column 7, line 26 * ---	1,2,6-8, 12,13,15	
A	KOVED K., NADALIN A. J., NEAL D., LAWSON T.: "THE EVOLUTION OF JAVA SECURITY" IBM SYSTEM JOURNAL, vol. 37, no. 3, 1998, pages 1-14, XP002144222 * page 4, line 1 - page 6, line 5 * -----	1,6,7	
			TECHNICAL FIELDS SEARCHED (Int.Cl.7)
The present search report has been drawn up for all claims			
Place of search THE HAGUE		Date of completion of the search 13 July 2001	Examiner Archontopoulos, E
<p>CATEGORY OF CITED DOCUMENTS</p> <p>X : particularly relevant if taken alone Y : particularly relevant if combined with another document of the same category A : technological background O : non-written disclosure P : intermediate document</p> <p>T : theory or principle underlying the invention E : earlier patent document, but published on, or after the filing date D : document cited in the application I : document cited for other reasons & : number of the same patent family, corresponding document</p>			

EPO FORM 1503 (03/02) (P4/C01)

ANNEX TO THE EUROPEAN SEARCH REPORT
ON EUROPEAN PATENT APPLICATION NO.

EP 01 41 0021

This annex lists the patent family members relating to the patent documents cited in the above-mentioned European search report. The members are as contained in the European Patent Office EDP file on
The European Patent Office is in no way liable for these particulars which are merely given for the purpose of information.

13-07-2001

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
WO 9916003 A	01-04-1999	US 6085229 A AU 9494198 A	04-07-2000 12-04-1999
US 5870544 A	09-02-1999	NONE	

EPO FORM PC-456

For more details about this annex : see Official Journal of the European Patent Office, No. 12/82